For Low Cost Computing
Pub. by
BILL HARMER
97 Ruskin St.,
Ottawa, Can. K1Y4B3

TS Bulletin #1-89

No, the TS Bulletin is not defunct! The delay between the December, 1988 and the current May issue, the first of 1989, marks a new policy of abandoning the attempt to produce a newsletter nearly every month of this title as well as starting to let the newsletters flow more naturally from inclination and interest. So the next issue you may receive from here may not be a TS Bulletin, but another newsletter substituted in the off months for the TS Bulletin. The publisher has some ideas on this already. The other thing that has put back the publishing schedule temporarily is work on a Larken TS1000 LDOS Disk Manual (now in draft stage and about two-thirds completed) and a manual for the TS1000 itself, for the first time user as well as those interested in technical specifications/explanations. These two unofficial manuals will be available through, it is planned, photocopying at user group libraries. Some of the material in them has already been published in this newsletter but most not.

In this issue the programming corner on page 2 deals with routines for using your TS1000 with a printer rather than a screen display. This has been mentioned in connection with the article on the TS1000 As A Portable Computer on page 7. An article on speculation on the next Sinclair computers is on page 3 and draws together material from numerous sources indicating that Sir Clive still has a trick or two up his sleeve. A background article on the early days of computers finds its way to page 4. It might be time to mention that a booklet collecting theee TS Bulletin articles is now available, under the title Famous Computers. The TS1000 Technical Manual and TS1000 Larken Disk Manual will be a while yet before they are ready for final release.

One more thing to mention is that things have been happening on both the development and supplier front. That is just one more reason to keep in touch with your user group and its library of newsletters, etc.

CONTENTS:

-The TS1000 As A Portable Computer p.7
-Routines For Using The TS1000 With Printer Only (Programming) p.2
-The Next Sinclair Computer (?). p.3
-With A Few Programming Tricks Up My Sleeve (BASIC Programming) p.5
-Ye Olden Golden Days Of High Technology (Early Computers) . . p.4
-Hobby Programming - A Type Of Miniature Modelling ? p.6
-The Great Debate Ends p.8

Update:
- delete the reference in the Larken El. ZX-81 disk manual to numeric array save and also that in the draft version of the Unofficial TS1000 LDOS manual.
- early versions of the ANT computer MS DOS emulator have some bugs. This is only one of the MS DOS emulators for the QL, but seems to be the cheaper one
- Vancouver TSUG reports unfilled orders placed with Jim Horne.* Most TS suppliers do it as a hobby now, note.

*mixed up order records blamed

This is a public domain newsletter published in order to encourage the idea that you don't have to be well off to enjoy the home computing hobby and in general to promote the extension of computer literacy using low cost home computers like the Timex 1000 and Sinclair ZX-81. It is published as a hobby activity by the writer and sent to a few user groups that support Timex-Sinclair computer users. Back issues are about $2 per issue. Warning: Check with a second qualified source before you hack, promise or buy using any information in this newsletter. (c) In the Public Domain, 1989

USING THE TS1000/ZX-81 AS A PORTABLE COMPUTER (WITH PRINTER AS OUTPUT)

There are certain tricks to using the TS1000/ZX-81 Timex-Sinclair computer as a portable. In some cases you may wish to run it without a television or monitor screen display. The best thing is to use a printer to give you feedback on what is happening and prompts on what to enter.

One problem is knowing when a program has finished loading. A routine like this will help:

```
9000 REM CASSETTE SAVE
9010 LPRINT "START CASS"
9020 PAUSE 300
9030 CLEAR
9040 SAVE "YRPROG"
9050 LPRINT "FINISHED"
9060 GOTO 1
```

This module will give you a prompt when it comes time to save the program and start the cassette recorder and it will tell when the save is finished. After a load too, the program will start up at line 9050 giving you notice that the load from cassette has been finished.

There is one other way of telling what your computer is doing and it works without a screen display or a printer. Holding a pocket AM radio with the antenna end (top) near various parts of the computer (and printer if it is attached) will give you different sounds at various stages in the process of cassette use. It will tell you when the loading or saving is going on, when it starts and stops and also it will tell you when you are near the end of a cassette save (the array loading sound similar to what you here when doing it with the volume turned on on the t.v.). There are even some loop programs (FOR...NEXT) that are designed to make different tones on the radio when you reach different parts of them (different looping speeds). Keep the magnet at the speaker away from tapes or disks though!

Speaking of prompts, there is a way to avoid too much extra programming and display both screen messages and printer prompts when using the computer without a television or monitor. The following is one suggestion:-

```
1000 REM CALLING PRINT ROUTINE
1010 LET D$="ENTER YOUR NAME NOW SO THAT IT CAN APPEAR IN
        THE DATABASE"
1020 GOSUB 9700
1030 ....
9700 Rem PRINT EITHER WAY SUBROUTINE
9705 LET D$=D$+"    <.<. 20 blank spaces> . "
9710 IF M$="T" THEN PRINT D$
9720 IF MS="P" THEN PRINT D$(1 TO 20)
9730 RETURN
...........
```

This will print on the printer when the M$ is set for T for television or P for printer,the appropriate message. It will only print the first 20 characters of the message on the printer to save wear and tear on it so you should make sure that all the essential information is on the first 20 character spaces of D$. Also, you must space the wording in D$ so that it breaks up into 32 character lines (with hyphens if necessary). Line 9705 prevents an error message and halt number 3 (subscript out of range) should D$ have fewer than 20 characters. If you don't want the extra spaces printed on the screen add a line 9725 LET D$=D$(1 TO ((LEN D$)-20)).

Another good thing is to echo input on the printer to make sure that you entered it correctly. INPUT A and then for the next line IF M$="P" THEN PRINT A. For listing a part of a program adding the line LLIST 1000 (if it is the module after line 1000 you wish to see) as line 9060 in the save routine above will cause it to list when it is loaded, if you do as I do and run your listings from a second TS1000 not connected to a screen.

All and all, a little thought and you will see a screen is optional.

The Next Sinclair Computer

Of course it won't be called a Sinclair. Sir Clive sold his name along with the rights to Spectrum and QL technology to Allan Sugar, head of Amstrad. So his last publicly released computer bears the name Cambridge, after Sir Clive's consulting company. But we are told that Sir Clive and his people are working on a new computer, a desktop this time (after the laptop Z-88). But no, it isn't the MS DOS desktop that you see in some ads with the Sinclair name on it. That is just an Amstrad with a 'Sinclair' name plate glued on it. No, this new desktop of Cambridge computers will be something special to judge from Sir Clive's track record of innovation in all the previous computers that bore his mark.

Some users are still using the ZX-81/TS1000 (like the writer). Many have moved up to the TS2068 often with a Spectrum ROM switch to make it compatible with many Spectrum programs available cheaply (for their degree of sophistication) from England. Programming languages like the C Language, Pascal and Prolog are available from one software house alone (HiSoft). Now disk drive interfaces such as that produced by Larken Electronics for the TS2068 (and as well a version for the ZX-81/TS1000, which the writer uses with his TS1000) are becoming much more common as Sinclair accessories. Aerco of Texas and Oliger also produce disk drive systems for the two computers. The QL users on the other hand, if they use a disk system, favour the Trump Card or Ace Card (both by Miracle Systems). The QL isn't seen much in Canada although a few have them, largely out west, and many have them in the U.S.A., but the greatest popularity of this computer is seen in Europe. The many games produced in England for the Spectrum and QL, are available with only the difficulty of making a direct mailorder purchase from ye old England and in addition many magazines in England and Europe that support the Sinclair line of computers are also available through overseas subscriptions. For sources of addresses, your local user group is the best place to start, and the library of magazines and newsletters the specific source usually.

But the question is what will Sir Clive do next? He has been dabbling in high technology, using the money from Amstrad's buyout of Sinclair, to commission work on wafer scale integration, a system that makes a computer on one wafer of silicon. The alternative is to make a bunch of integrated circuits on a wafer of silicon, cut them apart into small chips, test each one and take the good ones and put them into little plastic packages with metal legs that are then plugged into printed circuit boards to be connected together again electrically to make a computer. Wafer scale integration says,"Why cut the chips apart only to reconnect them again?" Why not put all the chips you need on one wafer and have them connected as part of the ordinary integrated circuit manufacturing process. The problem is now with this method,"What do you do with the bad areas which under the old system become bad chips that are thrown away, while the other chips cut from the wafer can be used?" The answer seems to be to make extra chips that are automatically bypassed by software in a computer running on the chip. This means that the computer on the chip must never be shut down (powered off) but then memory chips that always have a back-up battery connected to them in order to prevent loss of their contents are now pretty commonplace.

Sir Clive believes that with wafer scale integration one could produce a supercomputer that would fit in your pocket. Other possibilities for future projects by Sir Clive include parallel processing (by using multiple microprocessors designed to work together, called by one manufacturer Transputers), RISC microprocessors (Reduced Instruction Set Computers which speed up processing by allowing faster throughput of simpler machine language instructions), LISP or PROLOG-machine-language oriented-CPU computers for artificial intelligence, and so on. It is not beyond possibility that Sir Clive might market a computer in the future with a microprocessor of his own proprietary design. In addition, a RISC processor of sufficient speed could emulate any slower microprocessor through software, so that such a processor could emulate say an 8086 MS DOS microprocessor, a Z-80 microprocessor and a 68000 microprocessor, and run much of the code for such a computer (compiled programs, operating systems etc.), and be able to switch between them. Thus it could run all the Sinclair Z-80 based programs (and very fast) by emulating a Z-80, or run CP/M or MS DOS, or QDOS or ... any software that required a specific microprocessor to run on. The only problem would be configuring the rest of the computer to correspond to the features of the target computer to be emulated. That would be a massive software task. But once such a universal computer or microprocessor is launched, that could be left to contract or third party software developers.

At the moment we are left with the question of which way Sir Clive will go. Will he go for the relatively conventional or for the highly innovative? And a further question is if the eccentric genius will groom a successor company of innovators capable of providing economic and other continuity for follow through of what could be a long line of innovations. From ZX-80 to Z-88, Sir Clive has proven to be the computer Edison of our time, making the home computer as ubiquitous in British small businesses and households as Edison made the electric light. Hope can be held that the financial follow through will be there for innovation from Britain and that we will not see in Sir Clive's story yet another swan song of an innovative start not capitalized on. The British were right there at the start of computers in the 1940's as we know formerly secret information reveals. British designers in Canada built the Avro Arrow, a fighter 10 years ahead of its time. In both cases the early technical lead was lost to lack of practical and sound economic followthru. Will this be different? I hope so!

Ye Olden Golden Days of High Technology

In the real olden days of the 1940's, computers, then referred to in the newspapers as electronic brains (we know better now don't we?), were limited in access to the same number of people who today operate supercomputers. A more limiting fact that might be worth mentioning is that computers of that generation had only one terminal per computer, not many—multi-user, multi-tasking, time-sharing concepts all came later, much later (1970's typically). The real gas was that from the perspective of us hobby computerists, they were slower and had smaller memories than today's cheapest orphaned home computer. So, obviously the general public was out in the cold, non-computerized world for many a year, and had only the newspaper headlines on 'electronic brains', the science fiction stories (often projecting a future either patently impossible or not yet reached in computer capabilities in today's most advanced labs) and a few paperbacks that talked in generalities, to go by. These popularizations might mention Hollerith, Babbage, and Boole and that's it.

In the 1950's, technical books of all kinds were in short supply on the used market, and technical manuals on computers, the only useful way to learn about them were in the possession only of insiders, students in a computer school, working engineers, or computer operators or programmers. It was a rare person in even silicon valley who had access to the sort of technical information that we hobbyists take for granted. One of these who had the inside track was Steve Wozniak, whose father, a practicing engineer in silicon valley, one night brought home a briefcase full of IEEE technical specs for vacuum tube computers. The precocious child pored over them intently and after learning about vacuum tube storage devices possibly nixie tubes (incandescent number displays) and certainly vacuum tube flip-flops started on a journey of fascination with the lore of computers, leading to a DEC PDP-8 minicomputer manual, a part-time computer operation job and ultimately the Apple II. But before he designed the Apple II (and of course the earlier Apple I), he designed a dozen computers in his mind and scratch paper pads and dreams. Some teenagers draw cars. Steve drew computers and wiring diagrams for all sorts of little one-transistor (Germanium often then) gimmicks, designed to make a cricket noise or small radio transmitter.

What the designer of the Apple computer did then with the easily available info and plans from hobby magazines for radio buffs and the hard-to-find technical data for tube computers, was immerse himself in a technical culture that was mysterious to the layman, fascinating for the hobbyist and out of fashion at career days. The field to be in then was chemistry or chemical engineer. Electrical engineers were into radio and radar and not so much into computers. And university students had no opportunity (in the late 1940's and early 1950's) to even touch a computer unless they were studying to be an engineer or scientist. The field of computer science was still a branch of engineering, and a narrow specialty within it indeed, and an overlooked and underrated one at that.

The silicon chip put computer parts in the hands of the experimenter in the 1970's and the home computer came out for less than thousands of dollars each at about 1980 (ready-made rather than kit). Now every used book store of any size has a shelf or bin of the sort of technical books on computer science, computer parts, computer languages and operating systems that the avid seeker of the 1950's or '60's would have sold his little sister for (or worse). But since they are commonplace and cheap and getting the learning from them a task or a bore for many, they are the most underrated store of the history and course contents of a hundred technical colleges and universities (and the most ignored). You need not buy a $89 text or sign up for a $700 course or move to silicon valley to learn all you need to know to get started in the computer world. You need only to spend a couple of bucks on an old paperback on a once popular computer.

Perhaps it is a little oversimplifying to say that you can just learn it out of a book. But taking a couple of inexpensive courses at a local high school or college, in the evening, the summer or if you are of school age, as part of your regular curriculum will get you started. From then on a few books, some popular magazines and a cheap orphan computer bought at a flea market, out of a classified ad or from a computer user group swap day and you are on your way to computer literacy in a few short steps.

But I haunt the used book stores for a special reason. Computer history and technological history can only really be learned from the technical manuals of the period. Instruction sets and console diagrams are to us computer history buffs what the smell of black powder or picture of a civil war uniform are to the rest.

With A Few (Programming) Tricks Up My Sleeve

What are those tricks that the experienced programmer uses? How can one learn what they are? How can one learn to make them one's own?

Answer: If you do any programming now, you probably have few already! Take the first thing that you will want for nearly every program—a title page PRINT of the title on the screen, of course. A simple way of doing that is: PRINT " SAMPLE PROGRAM", then PRINT (to leave a blank line) and then whatever comes next in the program. A fancier way, reserving a whole page for the title, would be:

```
 10 PRINT AT 3,0;" ---------------" (enough for a whole line)
 20 PRINT AT 7,0;"       SAMPLE PROGRAM TITLE      " (Centered
                 program title within quotes)
 30 PRINT AT 11,0;" ---------------" (another whole line'-'s)
 40 FOR X=1 TO 200
 50 NEXT X
 60 CLS
 70 PRINT "PROGRAM TITLE AT TOP OF PAGE " (Use real title)
 80 PRINT
100 REM start main part of program here
```

There are many variations on this. Some computers use PRINT @ (one number) rather than PRINT AT (followed by two numbers). Some use LOCATE (and two or three numbers) and then PRINT. In the TS1000 such a LOCATE can be simulated by a PRINT AT (followed by two of the equivalent tho not necessarily same numbers as used in the locate statement) and just-;-with nothing after it. Then use a PRINT statement just as say GW BASIC in the IBM clones uses after LOCATE. But whether you use PRINT AT, PRINT @, LOCATE and then PRINT the trick of setting up a simple title page is one of the first in any programmer's bag of tricks. Once it is learned for one computer, using it in others is simply a matter of translation not reinventing the wheel. And that is the way a programmer's bag of tricks (or reusable routines, as it has recently become trendy to call them) is built up. You no doubt have ideas of your own but comparing notes may help too.

But what about the delay loop in lines 40 to 50. In different computers the number will vary to produce the same number of seconds delay (due to some computers being faster to go through a loop than others), but this will work for most computers. With the Sinclair/Timex TS1000 the command PAUSE (plus a number) can be used. PAUSE 60 would pause for 1 second, PAUSE 120 for two and so on. In computers of widely varying processor speeds but compatibility as to programs like IBM PC clones (that vary from 4.77 MHz to 20 MHz clocks for example), other methods than the FOR...NEXT loop can call on a built in TIMER command. If you are supplying a simple interpreted BASIC program, all the user has to do to fix it if the loop is too fast, is find it and change the FOR...NEXT loop to a higher number of loops (from 200 to 500 or 1000 for example).

But you may wish to use such a delay in more spots in your program than just the beginning title page. Therefore, why not put it in a subroutine at say, 5000 or so in BASIC line numbers, or at 8900 or something. Then a simple GOSUB 8900 say, would produce the delay and save on repeating the code over and over. The same routine could be used for longer delays, twice by calling it twice, for example.

Now we have two simple tricks from those up our sleeve. How about a third? Instead of just a delay, why not call up a "PRESS ENTER TO GO ON" routine like:

```
9300 REM TURN A PAGE ROUTINE
9310 PRINT AT 21,0;"PRESS ENTER TO GO ON"
9320 INPUT A$
9330 CLS
9340 RETURN
```

That is another trick and it has lots of variations including, "Press any key to go on" using INKEY$ and even flashing the 'PRESS' by printing it first in inverted letters (graphic) and then in ordinary, or using the facility of some computers to flash a message. Computers often can use continuous loops with GOTO inside of which is the PRINT commands and INKEY$. So once you have a borrowed or original bag of tricks, programming becomes easier no matter which computer you eventually switch to. The ideas are usually translatable if not the exact code.

Hobby Programming on the Home Computer -A Type of Miniature Modelling?

Miniature models while a source of fascination for all ages down through the years, from the miniature furniture in dolls' houses to the miniature cars and dump trucks of young boys' affections, can now take on a new form for the home computerist. That many things can be simulated or modeled on a computer is well known from movie colouration, animation and special graphic effects to games (zap-em and adventure) and even the biggest game for geniuses of our time, simulating our own human intelligence. But one of the most fascinating things for us is that the operation, look or feel of bigger or different computers can be simulated by our little homebased models.

One program for IBM-compatible computers allows you to write and assemble (convert to machine language) assembly language programs for the 1960's, 1970's vintage big IBM mainframe the IBM 360 (and IBM 370). Other programs allow you to write assembly language programs for use in microprocessor based electronic designs (like disk drive interfaces, appliance control units, etc.) and even convert between those programs written in the assembly language of one microprocessor to that in the assembly language of another. Other programs simulate the operator's look and feel in using another computer, minicomputer or another operating system (like the Wendin DEC VMS and the UNIX operating system simulation, for IBM compatible computers). So a number of programs are already out there in shareware or commercial software. What is less obvious is that most amateur programmers can with some time devoted to it write in the built-in BASIC of their computer similar simulations of other computers systems or even imaginary computers systems and programming or operator's environment, simulate the machine language or assembly language programming of another big computer or microcomputer or the computer on a space lander or Starvoyager. The main limitation with the smaller home computers is memory which determines how big a part of the system can be simulated or how much say the instruction set of an assembly language emulator need need be pared to make a containable subset. In addition, if you are able to get an assembler or a book of simple subroutines for your computer that has routines to do input, display, and even editing or simple integer math, writing at least a small emulator or simulator may not be beyond the range of a good machine language programmer. In my experience, 16 or 32K should suffice for a fairly involved simulator or emulator to be written in BASIC.

Say you want to emulate a simple microprocessor in BASIC. You will want some way of creating and editing programs. Why not use the array P$(X) to store a number of program lines, say up to 20 in the language, machine, assembly, or higher level that you want to make an interpreter for? The next thing is how to handle variables. If you are simulating a three register CPU, you might simply assign the variables RA, RB, and RC to represent the three registers. Then the instruction in your imaginary computer's assembly language would be CLEAR REGA. The interpreter would detect that the instruction is clear and that the operand (factor operated on) is REGA. Running it thru a table, IF OP$="REGA" AND CMD$="CLEAR" THEN LET RA=0, IF OP$="REGB" AND CMD$="CLEAR" THEN LET RB=0, and so on would be very simple. It would be preceded in the program by a routine to analyse the current command, and LET OP$= the current operand (for one-operand commands) and LET CMD$= the current command name. Running a program would simply take these instructions one after another as P$(X) and each time through the loop, the X would be incremented. A Branch instruction would simply plug a different value into the X variable, e.g. BRANCH BACKONE might subtract 1 from X and go through the command interpretation routine again (IF CMD$="BRANCH" AND OP$="BACKONE" THEN LET X=X-1 and then also GOTO instruction interpretation again. So writing a machine language or assembly language simulator/interpreter in BASIC is not all that hard. Addresses can be set up as labels and a search made down the label list (if the label of the current command is LAB$ then running through a table in an array, ALAB$(Y)(corresponding - INDX(Y)) so that for the first command the label START in the program (part of the instruction array element P$(X)) would be entered as ALAB$(1)="START", INDX(1)=1, so that you have the address label START mapped to the index 1, so that a BRANCH with an operand OP$="START" would take the index found by the label sort routine and LET X=it so that the next instruction would branch back to the beginning of the beginning of the program command list and making X=1 then the command to be analysed P$(X) would be P$(1). The possibility in some BASIC dialects of a GOTO Z or GOSUB Z allows you to call up subroutines in BASIC by a similar listing of labels in an array and numbers (which now represent BASIC line numbers, not indices of the array of instructions in the simulated machine language, P$(X)).

To summarize, it is possible to do things with BASIC that one would not thing possible before actually trying them. ROM calls using a label, (RAND USR Z), branches (GOTO Z) and subroutine calls within BASIC subroutines (GOSUB Z) are all possible directly in Timex-Sinclair BASIC's. In other BASIC's probably the same thing could be done by BRANCHING to a list of commands in BASIC. IF CMD$="CALL-MC" AND OP$="CLEARSCRN" THEN GOTO 4100 might take your to line 4100 GOSUB 1000 or maybe 4100 GOTO 990 or maybe even 4100 RAND USR 13762 and the next line 4110 might be a return to a reentry point like 4110 GOTO 560 which might be where the program goes after executing one instruction.

Using techniques like this you can invent your own microprocessor, in-house computer system, cross-assembler, minicomputer trainer, operating system trainer, machine language programming trainer, assembler, higher level language (MYPET Language, TOMMY's FORTRAN,e.g.) all in BASIC.

The ZX-81/TS1000 As A Portable Computer

There are a number of bargains currently on the market for portable computing use, from programmable calculators and pocket BASIC computers (like the Radio Shack PC-1, PC-2 etc.), through the Tandy 100, 102 and Cambridge Z-88, up to the heavier lugables like the Hyperion (MS DOS) and Osborne (CP/M), the latter with miniature picture tubes. But probably for less money, a surplus ZX-81 or TS1000 by Timex or Sinclair can fill in some of the functions of the lugables and even operate from (car) batteries in remote locations without power lines.

The first time that the writer saw a neat version of this was in the set up of a member of the Ottawa-Hull TSUG. This user had a ZX-81, RAM pack and TS2040 printer set up in an attache case, sort of loosely mounted in a transparent sheet plastic frame to space them out. Rather than use a television set for display (although a little 4 inch screen model would have been even neater) the programs were written to solicit data through prompts on the TS2040 thermal printer (LPRINT"ENTER EMPLOYEE NAME NOW" rather than PRINT"ENTER EMPLOYEE NAME NOW" for example). The system still needed something to supply the power (a wall plug), although it could also easily have been run off a car battery (outdoors, don't bring car batteries inside, where gasses can cause explosions). If a little 12 volt DC/120 volt AC convertible television set had been part of that, even that would run off a 12 volt car battery along with the other devices. Also to mention is that Timex itself once offered a custom carrying case similar in size to an attache case, for the TS1000. Too, the hard shell, aluminum cases for photo equipment or heavy duty attache cases for carrying repair persons' tools would also be good choices for cases for such portable applications.

To get technical for a moment (non-technclan might skip this paragraph), running a ZX-81/TS1000 from battery offers some technical challenges. Replacing the microprocessor Z-80A with a CMOS version might save a little power and running the computer as much as possible without its 16K or larger RAM pack would also be a big saving. Printers are big power hogs although most of the power is used during actual printing operations. Small televisions also use a fair amount of power and the ones with very tiny screens are almost useless (one and two inch screens). Further technical economies could be made by separating the power supplies internally. The least wasteful single battery would be a (big storage) battery with 9 volts output instead of 12 volts. (Note: the PC8300, a ZX-81 clone seems to need however, a full 12 volts to get clean cassette saves and loads). There are however two power rails in the TS1000/ZX one for 9 volts (and that might operate on as low as 7½ volts) and the other at 5 volts. In running the power at 9 volts or so though a 5 volt voltage regulator (7805) inside the computer, you are losing a fair amount of power. A 7805 requires a margin of 2 or more volts (depending on the internal resistance of the power supply, the less the better), over its output voltage of 5 volts. If you were to disconnect the output of the 7805 and connect the 5 volt rail that it would normally feed to a 6 volt battery with a more economical type of regulator such as several rectifier diodes in series, you might be able to get the computer working with some experimentation (if there are no big devices like RAM packs or printers sucking power from it especially at odd intervals such as when the printer goes to print). With a 6 volt battery driving the 5 volt supply through such a voltage reduction circuit and a 7½ or 9 volt battery driving the 9 volt rail separately, power could be conserved. Note that this would not work with a system using the old ZX-Printer (rather than the TS2040 or Alphacom 32). The ZX-Printer seems to need some sort of syncronization between the two power supplies and connection with the computer and drains the 5 volts.

One major hazzard of operating from batteries is getting the polarity reversed. There are several fixes. If running from a car battery, using a bridge rectifier permanently connected on the a.c. (~) terminals to battery clips and on the + and - side to the phone plug (+ to the tip ZX-81/TS1000, - to the shaft outside), it won't matter if you get the clips reversed on the battery, the computer will still work. Other safety devices for use with less robust batteries of lower voltage would include a series power supply diode wired into the supply cable between the clip leads and the plug or a crowbar diode in series with a fuse to the clip leads, and isolated from the plug to the computer by an on-off switch (with pilot light to make sure that you shut it off after using the computer).

One situation that a portable computer user might encounter in primitive conditions is not only running from battery but perhaps often running from power lines subject to frequent brown-out or black-out conditions. In that case a back-up power supply made up of enough penlight or flash light batteries to give 9 volts to operate the computer from for a minute or so until the program can be saved to a battery operated cassette deck, will save many problems. The cut in can be done using a couple of power supply diodes and some experimentation with your individual system's power requirements (see wiring diagrams in old magazines, newsletters, or SyncWare article reprint book). One such system, used by a member of our local user group has been quite successful. Of course the television would go blank in such a situation and it would not do to use the printer since it would tax the batteries, but if you could break the program to put it into immediate mode and enter, the command SAVE"X", a loss of work could be avoided.

So although the writer would not discourage you from buying other portables too, the ZX-81/TS 1000 might in a pinch serve as a portable as well and at a very reasonable cost.

The Great Debate Ends

The great debate in Timex-Sinclair circles has ended. The fundamentalists lost; the universalists won (at least a part of their platform). The MS DOS/IBM crowd snuck in by the back door when no one was watching it. Now, since the QL has a couple of software emulation packages to allow it to run programs that previously would require an IBM PC or clone, MS DOS is a part of the Sinclair community and it is no longer a heresy to discuss it in the newsletters or out.

Perhaps the failure in the defensive strategies was due to concentrating their energies on vilifying the MS DOS (and CP/M) computer manufacturer who bought Sir Clive's good name and started putting it on other computers that owed nothing to Sinclair's tradition of compatibility. They did not notice that CP/M had already sneaked in the back door by way of the Aerco TS2068, CP/M compatible RP/M operating system. Later CP/M sneaked in to QL computing by way of an emulator similar to that for MS DOS.

The QL emulators are the biggest news in the Sinclair User community in years. They are like the BASIC interpreters built into most home computers only instead of converting from BASIC statements to pull the strings via machine language subroutines of the home computer they convert the operating system hooks and eyes from MS DOS(loaded into the QL) into the appropriate machine language hooks and levers to actuate the hardware features of the QL.

The thing that seemed to convince Timex-Sinclair computer user groups to give MS DOS the cold shoulder was probably the unfortunate circumstances that led to the failure of Sinclair as an independent company and its take-over by Amstrad, a rival in the British home computer manufacturing field. The bitterness of the failure of Sinclair and the harsh but no doubt necessary terms of the takeover left a bad taste in the mouths of the Timex-Sinclair users who had been so enthusiastic in their praise of their little computers and of their designer-entrepreneur, Sir Clive Sinclair. The fear that the computers that even by then were losing support both from the manufacturer and third party companies, was riveting the Sinclair community with a growing apprehension and dread of what was to follow. The pruning of support for the broad-based (too broadly based at that time) Sinclair line did not make any friends for Amstrad, coming as it did at a time when so many other computer makes of similar class were being orphaned. Where are the MC-10 users, VIC-20, Coleco Adam suppliers now? At the time, we were just seeing what might be in store for Sinclair if support for it were to be eliminated at the manufacturer's level and it looked like our fears were starting to be realized.

Perhaps we should gloss over the indignant storm of protest that greated an article in Sinclair's SyncWare News magazine, a single article on MS DOS computers, comparing favourably the simplicity of the PC clones with the Sinclair computers. It was agonized over by editors of newsletters in the Sinclair community and dammned heartily by suppliers as helping to kick the skids out of the Sinclair suppliers' market.

Now as more and more Sinclair users are at least experimenting with MS DOS (and CP/M) via the use of software emulators, interest in MS DOS should be able to disassociate itself from overtones of disloyalty to Sinclair computing or/and defection to the IBM PC clone ranks.

For that reason as well as others, one may expect to see in the future a few articles in our newsletters dealing with MS DOS. It is no longer irrelevant to Sinclair computer users if it ever was. And please, dear reader, don't panic if you find one in these pages. It does not mean that we are hammering nails into the Sinclair coffin nor abandonning the Sinclair computers. It also need not imply that we are joining the ranks of Amstrad, MS DOS computer users.

As a bonus, at this time, the IBM clone/MS DOS computing world is entering a new period of uncertainty as to its future just as its range of public domain and shareware software has reached a breadth of subject coverage that no other computer has ever enjoyed. Disks selling for $5 or less provide as great a range as those available (in cassette form) for the ZX-81/TS1000 in its heyday. At the same time on the horizon the storm clouds are gathering and the flocks of business users are milling around, uncertain as to which new or updated version to go with. IBM AT clone, PS/2 up to model 40, PS/2 above model 50, OS/2, MS DOS, UNIX, all possibilities present the purchaser with a big quandary. The IBM computer got where it is today not simply by being good but even more by being popular and therefore exceedingly well supported in software and accessories as well as ironically by low cost cloning. If success depends on the number of computers installed and being used, it is pretty hard to predict which new system will be successful before trends in new computer and operating system installations are obvious. At the moment none is. But at least Sinclair users will be more than passive observers of the unfolding events, so exciting as to attract everyone's attention, news magazines, newspapers and sidewalk superintendents as well.